

System Security Engineering Capability Maturity Model and Evaluations: Partners Within the Assurance Framework

**Written by Charles G. Menk III
Department of Defense
June 1996**

Abstract:

Since the inception of the SSE-CMM program in 1993, there have been some misconceptions within the computer security and evaluation communities regarding its intended purpose. Evaluators in particular have expressed strong resistance to this effort due to the perception that the SSE-CMM is intended to replace evaluated assurance with developmental assurance. That has not and never will be the case. The SSE-CMM efforts can greatly enhance government, corporate, developer, user and integrator knowledge of security in general. As such, the efforts of the SSE-CMM development team are intended to provide significantly improved input to system developers (internal assessments) and the higher level assurance activities (e.g. evaluations, certification, accreditation) efforts (third party assessments). To best address the needs of our customers, the efforts of SSE-CMM and other assurance efforts must grow to complement each other. It will take focused effort from the security community and developmental assurance organizations, as well as industry partners to achieve this goal.

Evaluated assurance, provided by programs like the Trusted Product Evaluation Program (TPEP), has become widely accepted throughout the computer security industry. However, as the state of technology has advanced, the current process and methodology used by the evaluation community have been unable to keep pace with the accelerated development cycles of the advanced products that computer-security customers desire. The deficit of security expertise, as well as unclear and at times inadequate guidance and requirements within the industry and from government agencies has lead to the persistent practice among development organizations developing security as an afterthought or add-on to an existing product. Such practices make correcting security flaws that affect the underlying product expensive, difficult, and time-consuming. All of these factors have forced evaluators to carry out duties and activities far beyond the scope of pure evaluations and to take on the roles of trainer, developer, writer, and quality assurance inspector for the various products that they have been evaluating.

Given these sometimes conflicting demands on the evaluation process, it has become problematic if not impossible (in some cases) to expect the current evaluation approach to continue providing all the product security assurance and keep pace with the increasing demands of computer security customers (i.e. they can not produce enough evaluated products to meet the demand). That is where the concept of an Assurance Framework comes in. Each activity within the security arena (e.g. CMMs, ISO9000, Evaluations) brings with it a certain level of assurance. The composite view forms the Assurance Framework in which a customer can pick and choose products to support their mission based on their risk tolerance and product cost. By allowing certain activities, like the CMM efforts, to address specific assurance needs, the strain on the

evaluation community may be alleviated a little thereby allowing evaluators to focus on the high assurance products while the lower assurance products undergo a less rigorous assessment / certification process.

In the form of the SSE-CMM, developmental assurance can accomplish many needed improvements in the way that INFOSEC products and systems are produced. These improvements may well have a direct impact on the quality of the product's security development and can assist vendors by better preparing their teams for an evaluation. At the higher maturity levels, some of the work now required of evaluators for low assurance products, such as IV&V functions and general security knowledge, can be accomplished during the initial product development. This will allow evaluators to concentrate more of their efforts on evaluation activities and less on security education and or product development for the vendors. The SSE-CMM is a metric for an organization's capability to develop a secure system. Wouldn't it be nice to know an organization has the capability to build secure systems prior to accepting them into a rigorous evaluation activity?

At times, evaluation efforts seem to drag due to evaluator fears of missing critical security aspects and an overall miscommunication between the evaluation team and the developers. The evaluators' fears sometimes lead to "criteria creep" which is the desire to make a product the best it can be, within the criteria, instead of focusing on the requirements themselves and addressing the customers' needs. Miscommunication from the developers tend to result from an inability to clearly describe their product in security terms adequate to suffice the evaluation teams.

The SSE-CMM can help address both of these concerns. If evaluators know that a product is developed in a mature and security knowledgeable environment, they may be more at ease and willing to accept the developers inputs than they have been in the past. The developers, now aware and able to produce the evidence needed, may not feel that the evaluation process is as antagonistic and the communication lines will remain open, thereby enabling a smoother process and hopefully a more expeditious evaluation effort.

Using the SSE-CMM:

The ongoing development of the System Security Engineering CMM (SSE-CMM) has provided a new opportunity to revitalize government efforts to evaluate computer security systems on a timely basis. The understanding and application of the practices and principles embodied in the this model can help address many of the inefficiencies of the current post-development evaluation process. Below is a listing of the current Process Areas (PAs) of the SSE-CMM:

| PA Number | Title |
|-----------|----------------------------------|
| PA01 | Specify Security Needs |
| PA02 | Provide Security Input |
| PA03 | Verify and Validate Security |
| PA04 | Attack Security |
| PA05 | Assess Operational Security Risk |
| PA06 | Build Assurance Arguments |
| PA07 | Monitor System Security Posture |
| PA08 | Administer Security Controls |
| PA09 | Coordinate Security |
| PA10 | Determine Security Vulnerability |

The SSE-CMM can provide valuable insight in the following areas:

Engineering: A security engineering process improvement mechanism

Acquisition: A security engineering capability metric

Evaluation: A capability-based assurance mechanism

The SSE-CMM is designed as a tool for customers, developers and acquisition agencies to use in addressing their specific security needs. This paper will focus on the relationship between the SSE-CMM and evaluations. The three areas of concern for evaluations that we will highlight are System Planning, Development / Description and Testing.

System Planning:

There are two critical security issues that must be addressed early in the product life cycle to set the stage for the development of a Trusted product: policy and model definition and requirement analysis and definition. Historically, TPEP vendors provide policies derived after-the-fact. This leads to inconsistencies as they try to retro-fit the Policy and model to the operational system instead of building the system to address the policy and model specifications. The requirement analysis an implementation is often also done after-the-fact, resulting in last minute additions and modifications that may or may not be consistent with the policy and model. The result is a “dynamic” policy and model that may or may not be able to be evaluated due to it’s unstable nature.

Developing an accurate security policy and model for a system requires a well defined understanding of security policies and models, as well as a comprehensive knowledge of the system. Most weaknesses in this area result from an inadequate understanding of security modeling or a lack of commitment to security principles. Since the security model forms the foundation for the systems security, any problems with the model can cause cascading delays with the rest of the evaluation schedule while the vendor tries to correct the model and its implementation. Also, as more and more vendors hire third party contractors to write sections of code or purchase large portions (e.g. X Windows) of their system code, the developers are becoming less able to describe the system capabilities without third party intervention. This results in a disconnect between the policy, model and implementation that can not be readily addressed, causing further delays in the evaluations as the vendors attempt to elicit the information from the appropriate code and system developers and relay it back to the evaluators.

When security is done as an add-on, after development, there is a tendency to try to adapt and/or interpret the requirements to fit the product’s capabilities. This is usually the result of poorly defined security requirements that allow ambiguity and is sometimes driven by a wholly understandable self-interest on the part of the vendor (e.g. ROI decision process). This can cause recurring delays throughout an evaluation as evaluators have to argue and explain why the vendor’s interpretation and/or implementation of the requirement is not adequate to meet the evaluation criteria. The requirement analysis is further complicated when the basis for the analysis is an inadequate policy and model. In these situations, even a correct analysis would yield a failure to address the needs of the customer.

The SSE-CMM can greatly enhance a vendor's capability to address evaluation concerns by interjecting correct order in the system development life cycle. The Specify Security Needs (PA01) Process Area focuses on an organization's ability to capture the customer needs, identify applicable policies, laws, and constraints and identify the appropriate operating environment for the given system. Adherence to this and other process areas will enhance the developers' ability to define a system that addresses all the operational and functional requirements within the desired security specifications.

As part of the SSE-CMM, organizational processes that plan and track changes within the system are implemented. This capability meshes nicely with the evaluation Rating Maintenance Program (RAMP) process, which requires the tracking of security relevant changes to maintain the current rating for the product. Through the adherence to the SSE-CMM tracking requirements, a developer should be able to clearly articulate the changes and provide adequate evidence to support and expedite RAMP activity. This approach is preferable to the after market, add-hock patch-and-play approach used in many systems today because it gives a deterministic procedure for trouble shooting and analyzing the system to address the evaluation issues associated with maintaining a rating.

System Development

When vendors come to an evaluation process, they are expected to bring a complete Commercial Off The Shelf (COTS) system. The intent is that the evaluation group will receive a complete product, design documents and test suite from which they can verify the vendor claims with regard to the level of trust the product will operate under. Undocumented code makes design review, testing and verification efforts more difficult and in some cases untenable.

Malicious Insertion and Inadvertent Errors

The major concern about code within a secure system is whether it does exactly what it claims to do... no more and no less! Undocumented code that is inserted maliciously can lead to the most egregious security violations by adding hidden functionality to the system (e.g. trap doors, trojan horses, covert channels). Although supervisory oversight, peer reviews and shared knowledge of the product development can not prevent all violations, it will tend to reduce the likelihood of successful insertion and in many cases serves as an effective deterrent to such attempts. In addition to malicious attacks, there is always the potential for inadvertent errors. The effects of coding errors are most dangerous in an environment where they are tolerated or undetected. Sloppy development practices and poor reviews contribute to that environment and leave a system vulnerable to the effects of unintended activity. The possibility of hidden functionality and the potential for coding errors increases the time that an evaluator must spend analyzing and testing the system to ensure that it is secure.

The SSE-CMM can play a major role in preventing the likelihood of malicious insertions and inadvertent errors. The Provide Security Input (PA02) and Assess Security Risk (PA05) Process Areas measure an organization's ability to translate the customer needs into appropriate mechanisms within the system and monitor the development effort to ensure the project stays within the risk tolerances. Also, as an organization moves up the SSE-CMM scale, its capability

to conduct effective system and code reviews increases, the accountability is more structured and an effective peer review process is in place. These types of activities act as a strong deterrent to the malicious coder and can reduce inadvertent errors in the early stages of development thereby reducing the likelihood of failures being identified in testing and evaluations.

Design Documentation

Many security concerns are either not addressed in the standard documentation or are discussed in a vague, roundabout way. This produces delays in any evaluation effort because the existing documents must be revised or new documents developed to provide evaluators with the information that they need. These delays have a compounding effect because they occur after the evaluators and developers have already spent a significant amount of review time to determine the document's shortcomings.

Historically, the development and maintenance of documentation has been one of the toughest areas to keep current within a specific product development. This is mainly because developers are not tasked to write the supporting documents and may not communicate directly with the authors, which sometimes leads to confusion among the document writers resulting in contradictory or incorrect reflections of the system. When evaluators review this type of documentation, a significant amount of time is spent talking with the vendor to resolve the contradictions and correct the documents.

All of the issues discussed above with regard to documentation development and review lend to the high preponderance of late delivery of adequate security design documents to the evaluation body. In many cases the necessary information is not readily available to those responsible for describing the system specifications in written format. As a result, evaluation groups are faced with ongoing discussion of whether to believe the documents or the current system code. The question of whether the documents reflect what should be or are incorrect plagues evaluation groups and causes increased concern over the system's security posture.

Within the SSE-CMM, there is significant focus on the documentation produced throughout the System Security Engineering effort. These activities are encapsulated in the Assure Security (PA06) Process Area. As an organization's System Security Engineering capability maturity increases, the documents produced should provide a more clear and concise view of the security within the products or systems under development. This in turn will help evaluation efforts by reducing the number of document review iterations.

System Testing

Many products / systems today provide security as an afterthought, resulting in security-relevant interfaces and components being identified based on a review of the finished product. As such, the probability of missing a security-relevant item is greatly increased. This is often exacerbated by a disconnect between what vendors state as security relevant and what evaluators believe is security relevant. The result is added debate and extended schedules as the two "opposing" views are

worked through to a mutually acceptable compromise. With this approach, the security in any given product / system could be substantially different for each release as the results of negotiations become more subjective due to team composition and time constraints.

It has become more apparent that test suites are not complete when the evaluators are ready to test the system. The testing required for security evaluations is often beyond the scope of normal vendor testing. Therefore the appropriate tests must be created after the vendor decides to undergo an evaluation. Time has shown that vendors are capable of addressing their code, but testing breaks down when third party code, with little or no adequate design documentation or comment, is embedded in the product / system being evaluated. Incomplete test suites are fast becoming a recurring problem that leads to schedule slips that in some cases have placed entire evaluations on hold and in the worst cases resulted in evaluation termination.

In the rush to complete a security test suite, vendors who lack good development practices and quality controls may prepare tests that are not fully functional and in some cases, the output gives little indication of what the test actually accomplished (such as, "Test Passed"). This can force the evaluators to perform lengthy and tedious code reviews. In addition to wasting time in needless code reviews, other evaluation resources have been squandered when evaluators have been called to travel to a test site only to be told that they cannot do their job because the tests will not run. In some cases this failure has placed undue burden on evaluators to develop the right "vendor" tests (a function outside the scope of evaluations). Such failures stem from a lack of understanding the effects of security relevant changes to the base system. Many times vendors assume that a generic test suite will be sufficient to test a modified security platform and therefore no test verification is done, even though this should be a fundamental activity in the development cycle. Once again, we see the effects of reduced time lines and funding taking its toll on proper procedures. Had there been a mechanism in place to track the system security changes and their effects on the original tests, most vendors would be able to incorporate the appropriate fixes within their test suites prior to the actual testing cycle.

The SSE-CMM directly addresses the testing issues through the Verify and Validate (PA03) and Attack Security (PA04) Process Areas. Throughout the development effort, the SSE-CMM places great emphasis on communication between the various system engineering activities and the security activities. This should foster an environment more capable of integrating the security features into the main-line product. The result should be a well defined security / system interface and greater ability to develop complete and accurate test suites.

Conclusion:

Due to the rapidly changing, fast paced system engineering development cycles and increased demand for security, the customer demand for assurance, at a reasonable cost in time and money must be addressed. The SSE-CMM may not be the only solution, but it is a viable one.

Organizations that adhere to a documented, supported and mature security engineering process should be better able to define, build and deliver secure products on time and within budget. In addition to meeting the immediate needs in terms of timeliness and cost, an adherence to SSE-CMM practices should also provide a firm foundation for further, more in depth analysis (e.g. evaluations, accreditation and certification).

Now is the time to begin evolving evaluations and alternative options to ensure Commercial Off The Shelf (COTS) products that are state-of-the-art and secure. Since the RAMP program is so closely related to the CMM efforts, perhaps that relationship may provide fertile ground for the first linkages between developmental assurance and evaluations. The future of secure COTS solutions remains uncertain, but promising, if all the players (e.g. Government, Commercial, Evaluation and Alternate Assurance experts -- CMMs, ISO9000) are willing to work toward a common goal.

References:

SSE-CMM (Draft): TBD. Contact Charles G. Menk III for information: (410) 859-6091, menk@romulus.ncsc.mil.

SE-CMM: A Systems Engineering Capability Maturity Model (sm.), Version 1.1, November 1995, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213

SW-CMM: The Capability Maturity Model for Software, Version 1.1, February 1993, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213

TCSEC: Trusted Computer System Evaluation Criteria, DoD 5200.28-STD, December 1985, Department of Defense.

TNI: Trusted Network Interpretation, NCSC-TG-005, July 1987, National Computer Security Center, 9800 Savage Road, Fort Meade, MD 20755

TDI: Trusted Database Interpretation, NCSC-TG-021, Version-1, April 1991, National Computer Security Center, 9800 Savage Road, Fort Meade, MD 20755

RAMP: Rating Maintenance Phase Program Document, NCSC-TG-013, March 1995, National Computer Security Center, 9800 Savage Road, Fort Meade, MD 20755